



# Heuristics for flow shop rescheduling with mixed blocking constraints

Ayoub Tighazoui<sup>1</sup> · Christophe Sauvey<sup>2</sup> · Nathalie Sauer<sup>2</sup>

Received: 16 January 2023 / Accepted: 23 October 2023 / Published online: 20 November 2023  
© The Author(s) under exclusive licence to Sociedad de Estadística e Investigación Operativa 2023

## Abstract

In the flow shop rescheduling literature, many papers consider unlimited buffer capacities between successive machines. In real fact, these capacities may be limited, or no store may exist. Thus, a blocking situation is inducted. Diverse types of blocking constraints are studied in the flow shop scheduling problems. However, in dynamic environments, only few papers deal with these kinds of constraints. The aim of this paper is to investigate a problem of rescheduling the jobs in a flowshop environment and mixed blocking as a constraint, considering simultaneously schedule efficiency and stability as a performance measure, and job arrival as a disruption. An iterative methodology based on the predictive–reactive strategy is implemented for dealing with this rescheduling problem. The problem has first been modeled as a Mixed Integer Linear Programming (MILP) model. Experimental results show that the MILP resolution is only possible for small-sized instances. Hence, inspired by NEH algorithm, we proposed four heuristics for solving large-sized instances of this problem. Eventually, we discussed the performance of the proposed heuristics for different blocking situations, both in terms of solution efficiency and resolution time.

**Keywords** Rescheduling · Flow shop · Mixed blocking constraints · Heuristics · Efficiency · Stability

**Mathematics Subject Classification** 90B36

---

✉ Ayoub Tighazoui  
ayoub.tighazoui@unistra.fr

Christophe Sauvey  
christophe.sauvey@univ-lorraine.fr

Nathalie Sauer  
nathalie.sauer@univ-lorraine.fr

<sup>1</sup> Université de Strasbourg, ICUBE Laboratory, CNRS 7357, Strasbourg, France

<sup>2</sup> Université de Lorraine, LGIPM, 57000 Metz, France

## 1 Introduction and literature review

The adoption of digital technology in services companies has completely changed the customer's practices (Mula and Bogataj 2021). From now on, a customer can, in real time, create, modify, or cancel an order using new technological tools, such as mobile phone applications or companies' websites. To keep a competitive edge, the companies must be adapted to these orders uncertainties. In fact, the possibility to change the order's information has an impact on the production scheduling (Şenyiğit et al. 2022). Although traditional scheduling properly organizes the production system planning (Pitombeira-Neto and Prata 2020), this one will be invalid when a new disruption occurs. Therefore, the decision-makers must quickly react so as not to stop completely the production. As a consequence, the process of revising an existing schedule in response to disruptions, referred to as rescheduling, becomes a major issue for the enterprises, as well as an interesting research area.

Several papers have already established literature reviews of the rescheduling problems. Vieira et al. (2003) is among the well-known works that have provided a detailed definition of this process, and described its framework, methods, and strategies. The following works can also be noticed, Li et al. (2020), Serrano-Ruiz et al. (2021) and recently Mohan et al. (2022). In the scheduling literature, different types of machines environments have been studied, such as single machine representing a single workstation (Hall and Potts 2004; Prata et al. 2021), resources constrained (Machado-Dominguez et al. 2022), parallel machines representing identical or non-identical workstations (Kovalyov et al. 2019; Akyol Ozer and Sarac 2019; De La Vega et al. 2023), job shop (Yan et al. 2018; Serrano-Ruiz et al. 2022), flexible job Shop problem (Auer et al. 2021), and open shop (Liu and Zhou 2013; Ozolins 2021). Numerous papers are also dealing with flow shop rescheduling. Rahmani and Ramezani (2016) investigated a flexible flowshop rescheduling problem, with the arrival of new jobs as a disruption, the weighted tardiness as an efficiency measure, and the absolute deviation of completion times as a stability measure. The authors adopted a predictive–reactive rescheduling strategy and designed a variable neighborhood search algorithm to solve their problem. Katragjini et al. (2013) considered a permutation flow shop rescheduling problem. They assumed three simultaneous random disruptions, the arrival of new jobs, machine breakdowns, and release time delays. Makespan is considered as the schedule efficiency criterion, and the number of starting times altered tasks as the stability measure. The authors proposed an iterated greedy algorithm to solve the described problem. The following papers also present interesting studies about flow shop scheduling problems, Uhlmann et al. (2022) and Bautista-Valhondo and Alfaro-Pozo (2020).

The scheduling problems generally use classical efficiency criteria to assess the schedule performance. For instance, the makespan (Sayed et al. 2020; Valledor et al. 2022), the total completion time (Druetto et al. 2022), the total tardiness (Zhang et al. 2022a, b), the total weighted tardiness (Haroune et al. 2022), the number of tardy jobs (Zhang et al. 2013), and the multiple efficiency measures

(Kacem and Dammak 2021) are criteria that have already been studied. However, in dynamic environments, other measures can also be considered. Stability criteria, limiting the deviation from the initial schedule, are often associated with the efficiency criterion. They measure the penalty induced by moving jobs due to disruptions. In fact, the schedule movement generates supplementary costs, like reallocation costs, or raw materials reordering costs. Akkan (2015) considered a single machine rescheduling problem, disrupted by new jobs arrival. The authors associated the schedule efficiency and stability. They considered the maximum tardiness as an efficiency criterion, and the sum of the absolute starting time deviations as a stability measure, which is the sum of the absolute difference between the starting time of a job before and after the rescheduling. Other stability criteria have already been investigated in the rescheduling literature, for example the number of jobs that change of machine after rescheduling (He et al. 2020), change in human resources assignments (Xiao et al. 2010), and the maximum time disruption (delivery times of jobs to customers changes) (Liu and Ro 2014). Different from existing literature, this work investigates two performance measures. At first, regarding the schedule efficiency, the *Total Weighted Waiting Time (TWWT)* is used as a criterion. This measure is inspired from real-life systems, it represents the job waiting time in front of a workstation, or the patient waiting time in the case of hospital system (Braune et al. 2022), assuming the weight as the customer priority. Tighazoui et al. (2021a) considered a problem of the *TWWT* minimization combined with a stability criterion in a flowshop rescheduling problem. The authors explain that the problem can be an illustration of real-life systems. For example, in production systems, it can be regarded as the waiting period of a product in front of an installation and the weight as the product priority. In hospital systems, it can represent the delay between a patient's arrival and his actual treatment, considering the emergency level as a weight. Guo and Xie (2017) formulated two MILP models for a single machine rescheduling problem with the total waiting time as an objective. The studied problem came from a quartz glass factory, considering that the waiting time is the waiting of materials before the welding step. Kan (1976) classified the problem of the minimization of the average waiting time as NP-hard for a single machine scheduling. Second, concerning the schedule stability, the *Total Weighted Completion Time Deviation (TWCTD)* is considered as a criterion. It consists of assessing the deviation between the completion times of a job, when it is scheduled for the first time and after the rescheduling, associating a weight for each job, to penalize more largely the important orders (Tighazoui et al. 2020). This association of criteria has already been introduced in Tighazoui et al. (2021b) for the parallel machines rescheduling problem. The authors observed that due to a proactive effect, the stability measure provides better solutions compared to a mono criterion considering only the efficiency. Now, the aim is to analyze the criterion behavior in a flow shop rescheduling problem with blocking constraints.

Most of flow shop rescheduling problems assume that the buffer spaces capacities between successive machines are unlimited. In actual industrial systems, the store between successive machines is generally limited or no store space may exist. Thus, a blocking situation is generated, referred to as blocking constraint. In the

rescheduling literature, only few papers are interested in this question (Kecman et al. 2013; Tao and Liu 2019). Different types of blocking constraints have already been investigated in static flow shop scheduling. The classical *RSb* (Release when Starting Blocking) constraint for example has been introduced in Wang et al. (2006). It describes the case where a machine is blocked by a job, until the job starts on the following machine. *RCb* or *RCb\** (Release when Completing Blocking) describes the case where a machine is blocked by a job, until this one ends on the following machine (*RCb\**) (Yuan et al. 2009; Trabelsi et al. 2011) or starts on the further next one (*RCb*) (Martinez et al. 2006). Mixed blocking constraints represent the case where successive machines can be simultaneously subject to any combination of previous blocking constraints (Trabelsi et al. 2012). To fill the void, the present paper treats a flow shop rescheduling problem with different types of constraints mixed in one production system. The aim is to study a general case in which the decision-maker can choose the type of blocking constraint to apply between two successive machines. An example of mixed blocking constraints can be found in the cider brewing process. In this application, it is not possible to simultaneously brew apples for different customers. In this process, pouring the apples into the vat represents stage 1, grinding the apples is stage 2, and pressing the apples to make juice is stage 3. Between stage 1 and 2, there is an *RSb* constraint since all the apples must be poured before starting the grinding. Between stages 2 and 3, there is an *RCb* constraint since the apples of a new customer cannot be poured into the vat until all the apples of the first customer have been pressed.

Rescheduling strategies are classified in two basic categories (Herrmann 2006). *Dynamic scheduling strategy* consists in dispatching the jobs, using some rules at the moment of dispatching. In this strategy, heuristics are used to choose the sequence of the jobs that will be proceeded on the machine (Wu et al. 2022; Zhang et al. 2022a, b). *Predictive–reactive scheduling strategy* consists of generating an initial schedule at the first step, then updating it at each time a disruption occurs (Manzini et al. 2022). Gürel et al. (2010) used a predictive–reactive strategy on a parallel non-identical machines’ problem with controllable processing times. In the predictive phase, an initial schedule is generated to minimize the total manufacturing costs of the jobs. In the reactive phase, after a disruption caused by machines breakdowns, the schedule is repaired. In our paper, the predictive–reactive scheduling strategy is also adopted. In the predictive step, the flow shop scheduling problem with mixed blocking constraints is solved with the objective of minimizing the *TWWT*. After disruption occurrence, the reactive step starts, solving the rescheduling problem, but considering this time the *TWWT* as a criterion combined with the *TWCTD*. These two criteria are associated by the *efficiency–stability coefficient*  $\alpha$ , representing a ponderation between both parts of the objective function ( $\alpha TWWT + (1 - \alpha) TWCTD$ ).

The present work addresses a problem of rescheduling the jobs in a flowshop environment, mixed blocking as a constraint, and the arrival of the jobs as a disruption. A MILP formulation is designed to describe the problem, as well as an iterative methodology based on the predictive–reactive scheduling strategy. Experimental results show that the MILP resolution is only possible for little size instances. Hence, inspired by NEH algorithm, we proposed four heuristics for solving large

size instances of this problem. These heuristic methods have been compared to the resolution based on the MILP model, both in terms of accuracy and computation time. To the best of our knowledge, no other work was interested in studying this kind of problem. The main contributions of this paper are:

- Investigating for the first time a flow shop rescheduling problem with mixed blocking constraints.
- Simultaneously integrating the schedule efficiency and stability in the described problem.
- Designing a MILP model for this problem, and heuristic methods allowing to browse more jobs in a reasonable time.

The outline of the paper is as follows: Sect. 2 presents a description of the problem and the predictive–reactive strategy adopted to tackle it. Section 3 explains the MILP formulation of the offline problem, and the additional equation used to solve the online problem. In Sect. 4, heuristic methods are described. In Sect. 5, experimental results are presented and discussed. Finally, a conclusion and some perspectives are given in Sect. 6.

## 2 Problem description and methodology

In this section, the problem is described, as well as the predictive–reactive scheduling strategy proposed to handle the disruptions.

### 2.1 Problem description

A flow shop rescheduling problem with different types of blocking constraints is studied in this section. The system is also disrupted by the arrival of new jobs. To each job,  $j$  is associated a processing time on the machine  $m$ ,  $p_{jm}$ , a weight  $w_j$  and a release date  $r_j$ . Preemption is not assumed in this problem. Thus, when a job  $j$  starts its execution in the system, it proceeds up to the completion time  $CT_{jm}$ . Efficiency measure, stability measure, and blocking constraints are explained hereafter.

*The efficiency measure:*  $W_j$  is the waiting time of the job  $j$ . It represents the total period that the job  $j$  waits in the system before its complete execution.

In Fig. 1, the red lines represent the total period that job  $j$  waits in the system before its complete execution, it also represents the sum of the waiting periods that a job waits in front of the successive machines (workstations). Hence, the waiting time  $W_j$  is defined as:  $W_j = CT_{jnm} - r_j - \sum_{m=1}^{nm} p_{jm}$ , where  $CT_{jnm}$  is the job  $j$  completion time on the last machine.

It is assumed that  $N = \{1, 2, \dots, n\}$  is the set of jobs, and  $M = \{1, 2, \dots, nm\}$  is the set of machines. At time  $t=0$ , the set of jobs  $N$  are already available. Thus, the initial problem to solve before the disruptions consists in minimizing  $\sum_{j=1}^n w_j W_j$ , referred to as the *TWWT*.

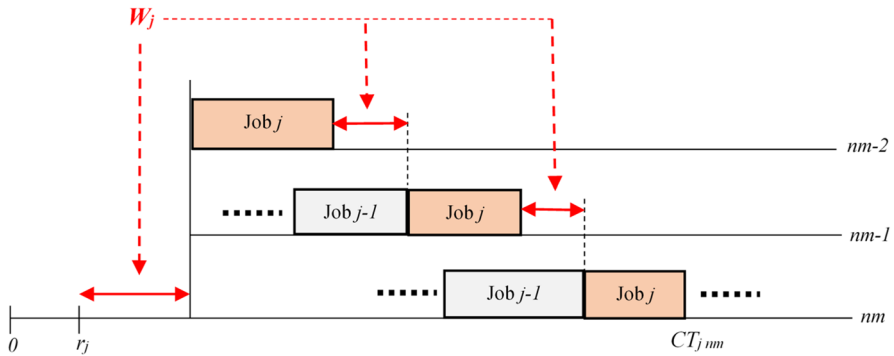


Fig. 1 Waiting time of a job *j*

The stability measure: let  $CT_{o_{jm}}$  be the original completion time of the job *j* on the machine *m*. It corresponds to the job *j* completion time when it is scheduled for the first time. After the occurrence of new jobs, the schedule may change, and the job *j* may then actually end at  $CT_{jm}$ , its actual completion time. Thus, the difference between  $CT_{jm}$  and  $CT_{o_{jm}}$  is the completion time deviation. In addition, the weight  $w_j$  is associated to each job *j* to penalize more largely the deviation of the important orders. In order to normalize this criterion with respect to the efficiency measure, it is divided by the number of machines  $nm$ . So, the stability measure is then defined as  $\frac{1}{nm} \sum_{j=1}^n \sum_{m=1}^{nm} w_j (CT_{jm} - CT_{o_{jm}})$ , referred to as the *TWCTD*.

Both parts of the objective function are associated with  $\alpha$ , the *efficiency–stability coefficient*. Thereby, the function simultaneously considers the schedule efficiency and the stability is:

$$\alpha \sum_{j=1}^{n'} w_j W_j + (1 - \alpha) \frac{1}{nm} \sum_{j=1}^n \sum_{m=1}^{nm} w_j (CT_{jm} - CT_{o_{jm}})$$

As can be seen, only the set *N* of jobs are concerned by the stability criterion. *N* contains the jobs existing before the occurrence of a disruption.  $N' = \{1, 2, \dots, n'\}$  is the set of jobs which contains the currently unexecuted jobs (i.e., the jobs that have not started) combined with the newly arrived ones.

The mixed blocking constraints: Most of flow shop rescheduling problems consider unlimited buffer space capacities between successive machines. This case is referred to as without blocking constraint (*Wb*). There is also the case of classical blocking constraints *RSb* when the machine is blocked by a job until this one starts in the following machine. Specific constraint *RCb\** can also be considered. It describes the case where a machine is blocked by a job until this one is finished on the following machine. *RCb* constraint is a particular case where a machine is blocked by a job, until this one is finished and left the next machine, this time then corresponds to the starting time of the job on the third machine. To illustrate the difference among these blocking constraints, we present in Fig. 2 an example with four jobs and five machines. The matrix of processing times is given by:

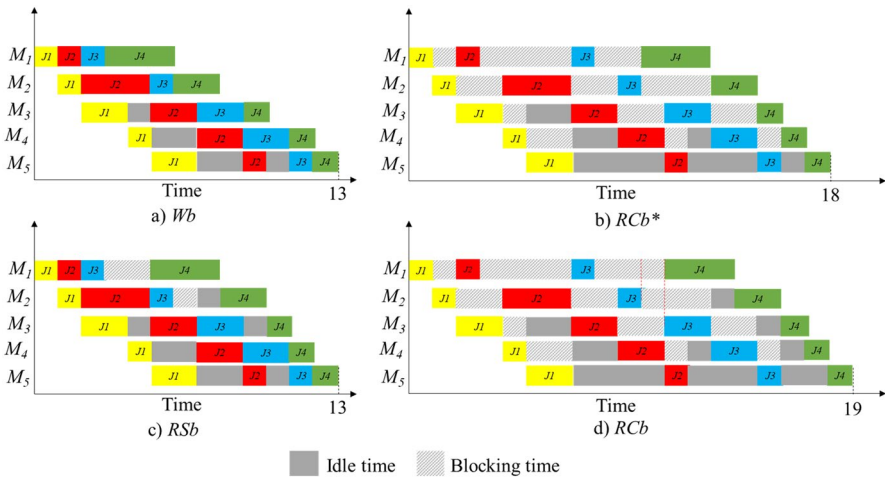


Fig. 2 Flowshop with different blocking constraints. a *Wb*, b *RCb\**, c *RSb*, d *RCb*

$$p_{jm} = \begin{pmatrix} 1 & 1 & 2 & 1 & 2 \\ 1 & 3 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 1 \\ 3 & 2 & 1 & 1 & 1 \end{pmatrix} j \in \{1, \dots, 4\}, m \in \{1, \dots, 5\}$$

In Fig. 2a, there is no blocking situation. Thus, each machine is available to treat a job as soon as the previous job is finished. In Fig. 2c, job  $J_3$  remains blocked on machine  $M_1$  while the machine  $M_2$  is executing job  $J_2$ . In Fig. 2b, illustrates the case when job  $J_3$  blocks the machine  $M_1$  until the operation  $J_3$  on  $M_2$  will finish. In Fig. 2d, illustrates the case when job  $J_3$  blocks the machine  $M_1$  until the operation  $J_3$  on  $M_2$  will finish and it will leave this machine. This date corresponds to the operation beginning of job  $J_3$  on the machine  $M_3$ .

In this work, we assume that all those described constraints can be mixed in one production system, with any combination. Thereby, a vector  $V$  is introduced, to describe a sequence of constraints between the successive machines of the problem. For instance,  $V = (RCb, RSb, RCb^*, Wb)$  can be a blocking constraints vector for the

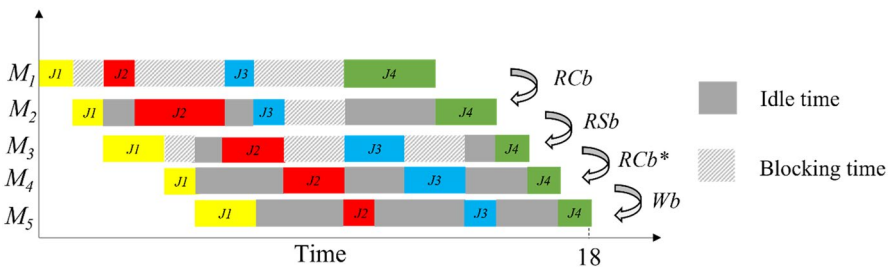


Fig. 3 Mixed blocking constraints with the vector  $V(RCb, RSb, RCb^*, Wb)$

5-machines problem presented on Fig. 3. Different combinations of the vector  $V$  are studied in the experimental results section.

## 2.2 Predictive-reactive scheduling strategy

In order to solve the flow shop rescheduling problem, we implemented a predictive–reactive scheduling strategy. It consists in solving at first, a problem of minimization of the  $TWWT$  efficiency measure. At the appearance of a new job, this one will be combined with the yet uninitiated jobs and the schedule will be updated. The new objective function contains a part of schedule efficiency measured with  $TWWT$ , as well as a part for assessing the schedule instability  $TWCTD$ . For simulating this matter, a finite time horizon  $[0, T]$  is defined, and discretized into periods  $\Delta t$ . The occurrence of a job may be possible only at these periods. It is assumed that  $\Delta t = 1$  unit of time, and one job can arrive per period. In several industrial cases, it is common to urgently treat the arrival of one order. When a job arrives at time  $t$ , this date will correspond to its release date. Accordingly, the variable  $\beta(t)$  is defined, equal to one if a job appears and zero otherwise.

$$\beta(t) = \begin{cases} 1 & \text{if a job appears} \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

The flowchart presented in Fig. 4 describes the proposed predictive–reactive scheduling strategy. This methodology consists of going through the simulation horizon step by step and checks whether a job appears thanks to the state of  $\beta(t)$ . If  $\beta(t) = 1$ , the problem data, constraints, and the objective function are updated. Then, the new problem is solved.

## 3 Mathematical models

According to the chart presented in Fig. 4, two phases of resolution are implemented. The first is before the disruption, referred to as the offline resolution. The second operates after the occurrence of the disruptions and is referred to as the online resolution. Accordingly, the two mathematical models corresponding to both phases are described in this section.

### 3.1 The offline mathematical formulation

To build our mathematical formulation, we adapted the parallel machine model presented in Tighazoui et al. (2020) to a flow shop environment with mixed blocking constraints. The presented model is based on assigning jobs to positions. Variables, parameters, and constraints are given hereafter:



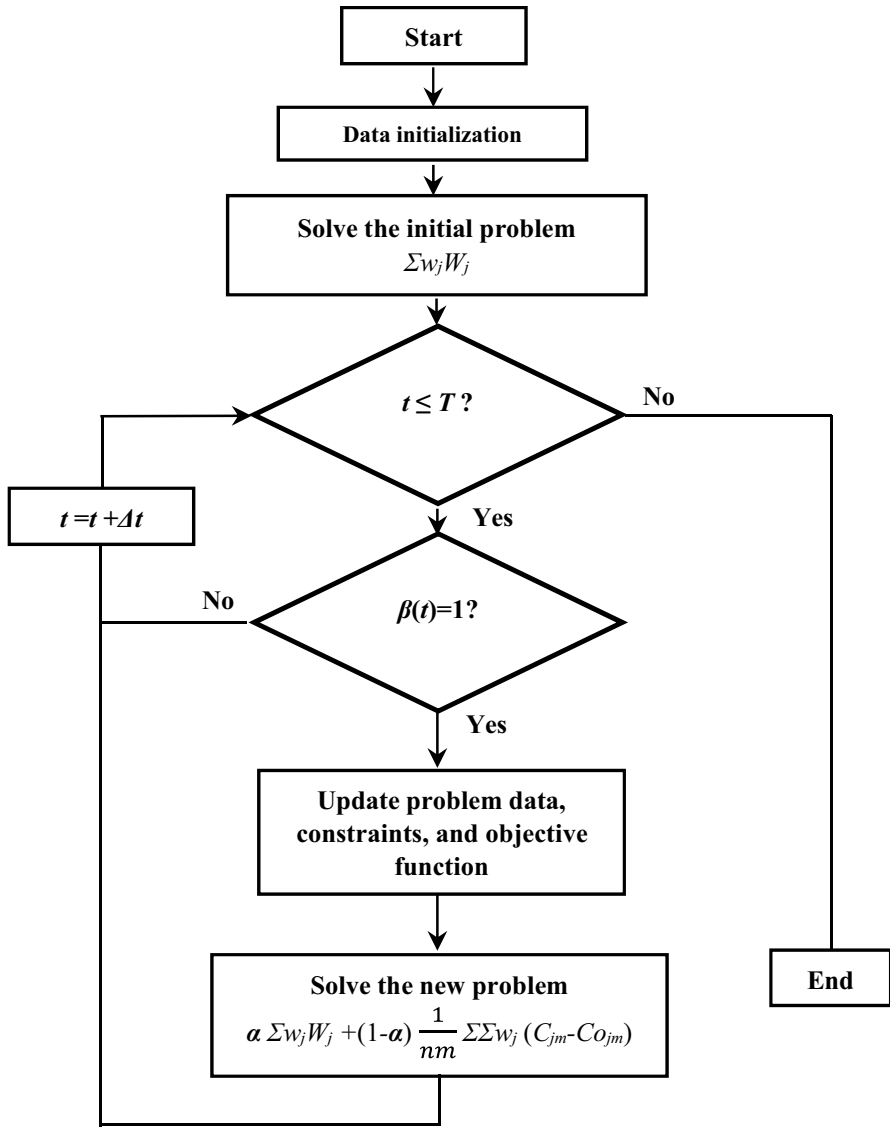


Fig. 4 Predictive-reactive scheduling strategy

### 3.1.1 Parameters

- $N$ : set of jobs  $\{1, 2, \dots, n\}$ .
- $K$ : set of positions  $\{1, 2, \dots, n\}$ .
- $M$ : set of machines  $\{1, 2, \dots, nm\}$ .
- $j$ : index of job,  $j = 1, 2, \dots, n$ .
- $k$ : index of position,  $k = 1, 2, \dots, n$ .

$m$ : index of machine,  $m = 1, 2, \dots, nm$ .  
 $w_j$ : weight of job  $j$ .  
 $r_j$ : The release date of job  $j$ .  
 $p_{jm}$ : processing time of job  $j$  on machine  $m$ .

$$B_{hm} = \begin{cases} 1 & \text{if there is a blocking constraint of type } h \text{ between the machine } m \text{ and } m + 1 \\ 0 & \text{Otherwise} \end{cases}$$

with:  $h = 1$  if there is no blocking constraint between the machine  $m$  and  $m + 1$ .  
 $h = 2$  if there is a blocking of type  $RSb$  between the machine  $m$  and  $m + 1$ .  
 $h = 3$  if there is a blocking of type  $RCb^*$  between the machine  $m$  and  $m + 1$ .  
 $h = 4$  if there is a blocking of type  $RCb$  between the machine  $m$  and  $m + 1$ .  
 $bigM$ : big value.

### 3.1.2 Decision variables

$$X_{jk} = \begin{cases} 1 & \text{if the job } j \text{ is assigned to } k\text{th position} \\ 0 & \text{Otherwise} \end{cases}$$

$S_{km}$ : Starting time of the job in  $k$ th position on machine  $m$ .

$C_{km}$ : Completion time of the job in  $k$ th position on machine  $m$ .

$CT_{jm}$ : Completion time of job  $j$  on machine  $m$ .

$W_j$ : Waiting time of job  $j$ .

Objective function  $f_1 = \sum_{j=1}^n w_j W_j$ , the objective is to Minimize  $f_1$  subject to:

$$\sum_{k=1}^n X_{jk} = 1 \quad \forall j \in N \tag{2}$$

$$\sum_{j=1}^n X_{jk} = 1 \quad \forall k \in K \tag{3}$$

$$C_{km} = S_{km} + \sum_{j=1}^n p_{jm} X_{jk} \quad \forall k \in K, \forall m \in M \tag{4}$$

$$S_{km} \geq C_{k m-1} \quad \forall k \in K, \forall m \in \{2, \dots, nm\} \tag{5}$$

$$S_{km} \geq C_{k-1 m} \cdot B_{1m} + S_{k-1 m+1} \cdot B_{2m} + C_{k-1 m+1} \cdot B_{3m} + S_{k-1 m+2} \cdot B_{4m} \tag{6}$$

$$\forall k \in \{2, \dots, n\}, \forall m \in \{1, \dots, nm - 2\}$$

$$S_{k nm-1} \geq C_{k-1 nm-1} \cdot B_{1 nm-1} + S_{k-1 nm} \cdot B_{2 nm-1} + C_{k-1 nm} \cdot B_{3 nm-1} \quad \forall k \in \{2, \dots, n\} \tag{7}$$

$$S_{k nm} \geq C_{k-1 nm} \quad \forall k \in \{2, \dots, n\} \tag{8}$$

$$S_{km} \geq \sum_{j=1}^n r_j X_{jk} \quad \forall k \in K, \forall m \in M \tag{9}$$

$$CT_{jm} \geq C_{km} - bigM(1 - X_{jk}) \quad \forall j \in N, k \in K, \forall m \in M \tag{10}$$

$$CT_{jm} \leq C_{km} + bigM(1 - X_{jk}) \quad \forall j \in N, k \in K, \forall m \in M \tag{11}$$

$$W_j = CT_{jnm} - r_j - \sum_{m=1}^{nm} p_{jm} \quad \forall j \in N \tag{12}$$

$$X_{jk} \in \{0, 1\}, \quad \forall j \in N, \forall k \in K \tag{13}$$

$$S_{km}, C_{km}, CT_{jm}, W_j \geq 0 \quad \forall j \in N, k \in K, \forall m \in M \tag{14}$$

Constraint (2) specifies that each job is assigned to one position. Constraint (3) specifies that each position is occupied by only one job. Constraint (4) defines the completion time of the position  $k$ . Constraint (5) an operation on a machine starts after its completion on the previous machine. Constraint (6) defines the starting time of a position. It depends on different types of blocking constraints, depending on the parameter  $B_{hm}$ . Constraint (7) defines the starting time of a position in the penultimate machine where there is no  $RCb$  constraint, as it depends on the two following machines. Constraint (8) defines the starting time in the last machine which can only be without blocking. Constraint (9) makes the starting time of an operation greater than or equal its release date. Constraints (10) and (11) associates the job completion time to the position completion time, where  $bigM$  must be sufficiently large. Constraint (12) defines the waiting time of job  $j$ , according to its completion time in the last machine, release date, and processing time. Constraint (13) constraints the variable  $X_{jk}$  to be a binary decision variable. Constraints (14) are non-negativity constraints, making all decision variables greater than or equal to zero.

### 3.2 Online mathematical formulation

This second mathematical formulation is used after a disruption. Thus, all jobs that have already begun their execution before  $t$  (the time of disruption) are omitted from the set of jobs. The new parameters are presented hereafter:

#### 3.2.1 Parameters of the online mathematical formulation

$N_j$ : number of new jobs (we assumed in this work that only one job arrives per period  $n_j = 1$ ).

$nt$ : number of unexecuted jobs at time  $t$ .

$n' = nt + n_j$

$N'$ : set of jobs  $\{1, 2, \dots, n'\}$ .

$K'$ : set of positions  $\{1, 2, \dots, n'\}$ .

$j$ : index of job  $j = 1, 2, \dots, n'$ .

$k$ : index of position  $k = 1, 2, \dots, n'$ .

$CTo_{jm}$ : original completion time of job  $j$  on machine  $m$ .

$\alpha$ : efficiency–stability coefficient.

The same decision variables as previously described are used for this second model. Thus, the new objective function is defined as:

$$f_2 = \alpha \sum_{j=1}^{n'} w_j W_j + (1 - \alpha) \frac{1}{nm} \sum_{j=1}^n \sum_{m=1}^{nm} w_j (CT_{jm} - CTo_{jm})$$

## 4 Heuristic methods

The proposed heuristics are based on the initial solution to insert the new job on schedule. The heuristics browse in the simulation horizon step by step, at each step  $t$ , if  $\beta(t) = 1$ , a job arrives. This job will be combined with the set of unexecuted jobs, its release date will correspond to the time  $t$ , and the last position in the existing schedule will be assigned to it. Then, one of the proposed heuristic methods will be applied. The following algorithm describes this process.

For  $t = 1$  to  $T$

  If ( $\beta(t) = 1$ )

$n' = nt + 1$                    ! add the new job to the set of unexecuted jobs

$r(n') = t$                    ! the time  $t$  is assigned to the release date of the new job

    Put the new job in the last position.

    Apply one of the proposed heuristics.

  End if

End for

The proposed heuristics are based on the NEH heuristic. This latter is one of the best well known in the scheduling problem since its effectiveness is proven (Nawaz et al. 1983). It is also very simple to implement. NEH algorithm is widely accepted as one of the best simple heuristics for makespan minimization in flow shop scheduling problems (Sauvey and Sauer 2020). As NEH is based on job insertion, it is wise to use it in the case of a disruption due to job arrival, since the new job is

inserted into the existing schedule. In our case, we use NEH algorithm evaluating the new objective function  $f_2$ . NEH pseudo-code is recalled below:

```

Rank jobs in decreasing order of total processing time (initial sequence)
Remove first job of ranked list and insert it as first element of partial sequence
 $k = 1$ 
Do
    Take first job of ranked list
    Insert it in all of  $k + 1$  possible positions of current partial sequence
    Evaluate all of  $k + 1$  resulting partial sequences
    Keep the best sequence as new current partial sequence
     $k = k + 1$ 
While ( $k \leq n'$ )

```

#### 4.1 Heuristic H1

**Description of H1:** We consider the list of jobs sequenced in the order of the previous schedule and we insert the new job in all possible positions, starting by the last position. The solution that minimizes the objective function is selected.

**Algorithm H1:**

```

Best_objective =  $\infty$ 
 $k = n'$ 
While ( $k > 0$ ) do
    Put the new job in the position  $k$  and shift to the right the jobs starting from this position without changing their respective orders.
    Objective = value of the objective function of this schedule
    If (Objective < Best_objective)
        Select this solution.
        Best_objective = Objective
    End if
     $k = k - 1$ 
End while

```

**Improvement of H1:** We consider the sequence obtained by H1 and we put the last job in all possible positions. We select the solution if it is better. We repeat the same operation, while the value of the objective function is improved. We call this heuristic H1\*.

**Algorithm H1\*:**

Let  $S$  the sequence obtained by H1.  
 $Best\_objective$  = objective obtained with  $S$ .  
 $Objective = 0$   
 While ( $Objective \leq Best\_objective$ ) do  
   Apply H1 from  $S$ , acting on the last job in the sequence (The last job is inserted at all possible position) and obtain the best sequence  $S1$   
    $Objective$  = objective obtained with  $S1$   
   If ( $Objective < Best\_objective$ )  
      $S = S1$   
      $Best\_objective = Objective$   
 End if  
 End while

**Example illustrating H1 and H1\*:** In Table 1, the processing times  $p_{jm}$ , the release dates  $r_j$  and the weights  $w_j$  for the 4 jobs are presented. The first 3 jobs are considered in the initial schedule (in *offline*), for which all information is known in advance, then job 4 arrives at  $t=3$  (in *online*). There is no blocking constraints between the machines  $V(Wb, Wb)$ , and  $\alpha=0.5$ .

Optimal initial sequence is 1-2-3 with  $f_1=15$ . Table 2 presents the obtained solutions with H1 and H1\* after the apparition of job 4.

H1 placed job 4 in all possible positions, except for the first one since the job 1 has already started its execution when job 4 arrived at time  $t=3$ . The best solution provided by H1 is  $f_2=19.83$  instead of 18.5 given by H1\*, which improves H1 by repeating the same operations, as long as a better result is obtained. However, the computing time of H1\* is bigger than H1.

**4.2 Heuristic H2**

**Description of H2:** We consider a list of jobs sequenced by the order obtained in the previous schedule and put temporarily the new job in the last position. We apply NEH for sequencing this list. The solution that minimizes the objective function is selected.

**Table 1** Problem data of the illustrative example

	Offline			Online
Job $j$	1	2	3	4
$p_{jm}$				
$M_1$	3	1	5	1
$M_2$	2	4	2	2
$M_3$	3	2	4	3
$r_j$	0	1	0	3
$w_j$	2	1	3	5

**Table 2** Obtained solutions with H1 and H1\*

	H1		H1*	
	Sequence	$f_2$	Sequence	$f_2$
First execution				
Iteration 1	1-2-3-4	30	1-2-3-4	30
Iteration 2	1-2-4-3	27.5	1-2-4-3	27.5
Iteration 3	1-4-2-3	<b>19.83</b>	1-4-2-3	<b>19.83</b>
Improvement 1				
Initial solution	-	-	1-4-2-3	19.83
Iteration 2	-	-	1-4-3-2	<b>18.5</b>
Iteration 3	-	-	1-3-4-2	32
Improvement 2				
Initial solution	-	-	1-4-3-2	<b>18.5</b>
Iteration 2	-	-	1-4-2-3	19.83
Iteration 3	-	-	1-2-4-3	27.5
Best solution	1-4-2-3	19.83	1-4-3-2	<b>18.5</b>
Computing time (s)	0.046		0.069	

The values highlighted in bold represent the optimal outcomes achieved in each iteration

**Improvement of H2:** We consider the sequence obtained by H2 and apply the algorithm of NEH recursively as for H1\*, while an improvement is found. We call this heuristic H2\*.

**Example illustrating H2 and H2\*:** We study here an illustration of H2 and H2\*, with the same example as the one presented in Table 1. The obtained results are presented in Table 3.

H2 considers the list 1-2-3-4. Exempt the jobs that already started their execution (job 1 in this example), H2 puts the jobs of the list, one by one, in the partial schedule applying NEH method. The best solution provided by H2 is  $f_2=19.83$  instead of 18.5 given by H2\*, which improves H2 by repeating the same operations as long as the result is better. However, the computing time of H2\* is bigger than H2.

### 4.3 Heuristic H3

**Description of H3:** We consider a list of jobs sequenced by the order obtained in the previous schedule and put temporarily the new job in the first position. We apply NEH to sequence this list. The solution that minimizes the objective function is selected.

**Table 3** Obtained solutions with H2 and H2\*

	H2		H2*	
	Sequence	$f_2$	Sequence	$f_2$
Execution of heuristic				
Iteration 1	1-2	1.5	1-2	1.5
Iteration 2	1-2-3	7.5	1-2-3	7.5
Iteration 3	1-3-2	9.5	1-3-2	9.5
Iteration 4	1-2-3-4	30	1-2-3-4	30
Iteration 5	1-2-4-3	27.5	1-2-4-3	27.5
Iteration 6	1-4-2-3	19.83	1-4-2-3	<b>19.83</b>
Improvement 1				
Iteration 1	-	-	1-4	5
Iteration 2	-	-	1-4-2	8.33
Iteration 3	-	-	1-2-4	14
Iteration 4	-	-	1-4-2-3	19.83
Iteration 5	-	-	1-4-3-2	<b>18.5</b>
Iteration 6	-	-	1-3-4-2	32
Improvement 2				
Iteration 1	-	-	1-4	5
Iteration 2	-	-	1-4-3	11
Iteration 3	-	-	1-3-4	23
Iteration 4	-	-	1-4-3-2	<b>18.5</b>
Iteration 5	-	-	1-4-2-3	19.83
Iteration 6	-	-	1-2-4-3	27.5
Best solution	1-4-2-3	19.83	1-4-3-2	<b>18.5</b>
Computing time (s)	0.047		0.078	

The values highlighted in bold represent the optimal outcomes achieved in each iteration

**Improvement of H3:** We consider the sequence obtained by H3 and apply the algorithm of NEH recursively as for H1\*, while an improvement is found. We call this heuristic H3\*.

**Example illustrating H3 and H3\*:** We study here an illustration of H3 and H3\*, with the same example as the one presented in Table 1. The obtained results are presented in Table 4.

H3 considers the list 1-4-2-3. Exempt the jobs that already started their execution (job 1 in this example), H3 puts the jobs of the list, one by one, in the partial schedule applying NEH method. The best solution provided by H3 and H3\* is  $f_2 = 18.5$ . However, the computing time of H3\* is bigger than H3.



**Table 4** Obtained solutions with H3 and H3\*

	H3		H3*	
	Sequence	$f_2$	Sequence	$f_2$
Execution of heuristic				
Iteration 1	1-4	5	1-4	5
Iteration 2	1-4-2	8.33	1-4-2	8.33
Iteration 3	1-2-4	14	1-2-4	14
Iteration 4	1-4-2-3	19.83	1-4-2-3	19.83
Iteration 5	1-4-3-2	<b>18.5</b>	1-4-3-2	<b>18.5</b>
Iteration 6	1-3-4-2	32	1-3-4-2	32
Improvement 1				
Iteration 1	-	-	1-4	5
Iteration 2	-	-	1-4-3	11
Iteration 3	-	-	1-3-4	23
Iteration 4	-	-	1-4-3-2	<b>18.5</b>
Iteration 5	-	-	1-4-2-3	19.83
Iteration 6	-	-	1-2-4-3	27.5
Best solution	1-4-3-2	<b>18.5</b>	1-4-3-2	<b>18.5</b>
Computing time (s)	0.047		0.063	

The values highlighted in bold represent the optimal outcomes achieved in each iteration

**Table 5** Obtained solutions with H4 and H4\*

	H4		H4*	
	Sequence	$f_2$	Sequence	$f_2$
Execution of heuristic				
Iteration 1	1-4	5	1-4	5
Iteration 2	1-4-3	11	1-4-3	11
Iteration 3	1-3-4	23	1-3-4	23
Iteration 4	1-4-3-2	<b>18.5</b>	1-4-3-2	<b>18.5</b>
Iteration 5	1-4-2-3	19.83	1-4-2-3	19.83
Iteration 6	1-2-4-3	27.5	1-2-4-3	27.5
Improvement 1				
Iteration 1	-	-	1-4	5
Iteration 2	-	-	1-4-3	11
Iteration 3	-	-	1-3-4	23
Iteration 4	-	-	1-4-3-2	<b>18.5</b>
Iteration 5	-	-	1-4-2-3	19.83
Iteration 6	-	-	1-2-4-3	27.5
Best solution	1-4-3-2	<b>18.5</b>	1-4-3-2	<b>18.5</b>
Computing time (s)	0.048		0.062	

The values highlighted in bold represent the optimal outcomes achieved in each iteration

#### 4.4 Heuristic H4

**Description of H4:** We consider the list of jobs sequenced by descending order of  $w_j$  and we apply NEH to sequence this list. The solution that minimizes the objective function is selected.

**Improvement of H4:** We consider the sequence obtained by H4 and apply the algorithm of NEH recursively as for H1\*, while an improvement is found. We call this heuristic H4\*.

**Example illustrating H4 and H4\*:** We study here an illustration of H4 and H4\*, with the same example as the one presented in Table 1. The obtained results are presented in Table 5.

H4 considers the list of jobs sequenced by descending order of  $w_j$ , 4-3-1-2. Exempt the jobs that already started their execution (job 1 in this example), H4 puts the jobs of the list, one by one, in the partial schedule applying NEH method. The best solution provided by H4 and H4\* is  $f_2 = 18.5$ . However, the computing time of H4\* is bigger than H4.

## 5 Experimental results

In this section, numerical results obtained for a flow shop system composed by 5 machines are presented. The data used are explained in Table 6.

The chosen data are adapted for real industrial cases. The simulation is over 8-h' time horizon (480 min), representing a factory opening time. It is assumed that  $\Delta t = 1$  unit of time (*ut*), and 1 *ut* is equivalent to 10 min. Thus,  $T = 8 \text{ h} = 480 \text{ min} = 48 \text{ ut}$ . The weight values  $w_j$  represent 5 priority levels of customers. The processing times  $p_{jm}$  represent the durations of product manufacturing times. It is assumed that it follows a discrete uniform distribution with values between 1 and 4, obtaining durations between 1 *ut* (10 min) and 4 *ut* (40 min). In the offline phase, the release dates are generated following a discrete uniform distribution, providing values between 0 and 2,  $U \sim (0, 2)$ , to have an initial solution on which we have generated the disruptions. In the online phase, the variable  $\beta(t)$  is used for generating the arrival of new jobs, its value is randomly generated with the Bernoulli distribution. At each time  $t$ , the value 1 is generated with probability  $p_\beta$ , and 0 with probability  $1 - p_\beta$ , where  $p_\beta$  is the appearance frequency of the jobs. Different values of  $p_\beta$  are tested.

50 different instances are randomly generated according to  $p_{jm}$  and  $w_j$ . For each instance, we scheduled the first five jobs in the offline phase, considering

**Table 6** Parameter values

Parameters	Values
$T$	48 ( <i>ut</i> )
$w_j$	$\sim U(1, 5)$
$p_{jm}$	$\sim U(1, 4)$ ( <i>ut</i> )
$\beta(t)$	$\sim B(p_\beta)$

that the information is known at time  $t=0$ . Then, the schedule will be disrupted by the arrival of new jobs. For each instance, we tested three scenarios of  $\alpha$  (0.5, 0.75 and 1) and four scenarios of  $p_\beta$  (0.2, 0.5, 0.8 and 1). In total, we investigated  $50 \times 3 \times 4 = 600$  scenarios. Thus, three vectors of constraints are studied, in order to analyze the impact of the blocking constraints on the solution.

## 5.1 Without blocking constraints case

In this case, a vector without any blocking constraint  $V(Wb, Wb, Wb, Wb)$  is considered. It represents a classical flowshop rescheduling problem. Two studies are conducted in this section. We firstly made a comparison between the proposed heuristics and their improved versions, to quantify the improvement process, both in terms of efficiency and computing time. Secondly, the heuristics are compared with the method based on the MILP model (B-MILP).

### 5.1.1 Heuristics versus improved heuristics

A comparison of the heuristics ( $H$ ) and their improved versions ( $H^*$ ) has been established. Ten instances are generated, accordingly to Table 6. For each heuristic, the Number of Times when  $H^* \leq H$  ( $NT$ ), the Improvement Rate ( $IR$ ), and the Time Difference Rate ( $TDR$ ) are calculated. Averages are presented in Table 7.

$$IR = \frac{\text{Solution given by } H - \text{Solution given by } H^*}{\text{Solution given by } H}$$

$$TDR = \frac{\text{Time of } H^* - \text{Time of } H}{\text{Time of } H}$$

The improvement rate is, in most of cases, positive. That proves the efficiency of the improved versions of the heuristics. However, in some particular cases, this rate can be negative. By applying the improvement on some steps, we certainly get a better solution. But, in the next step, the methodology consists in fixing the already executed jobs in the previous step and rescheduling. In this case, if the set of jobs fixed by  $H$  and  $H^*$  is not the same, the problem to solve in the next steps becomes different. Thus, the improved version can provide a bad result in the final step, but the value of  $IR$  when it is negative is relatively small. On the other hand, the average of the number of times when  $H^* \leq H$  ( $NT$ ) is 7.8.

Time difference rate is always positive, since the improvement consists of repeating the operations of the heuristic if the solution is better. Thus, the computing time increases. Thanks to their effectiveness, in the rest of the experimental results, the improved heuristics are used for the resolution methods comparison.

**Table 7** Comparison between the heuristics and their improved versions

	$p_\beta=0.2$ (10 jobs)				$p_\beta=0.5$ (24 jobs)				$p_\beta=0.8$ (39 jobs)				$p_\beta=1$ (48 jobs)			
	H1*	H2*	H3*	H4*	H1*	H2*	H3*	H4*	H1*	H2*	H3*	H4*	H1*	H2*	H3*	H4*
	$\alpha=1$	IR	0.009	0.001	0.027	0.000	-0.006	0.019	0.000	0.020	0.000	0.006	0.015	0.000	0.004	0.010
	TDR	0.174	0.369	0.445	0.509	3.138	1.073	1.092	1.081	14.055	1.350	1.374	22.674	1.510	1.153	1.198
	NT	10	10	10	10	7	8	8	10	7	7	9	8	8	9	9
$\alpha=0.75$	IR	0.005	0.000	0.000	0.000	0.003	0.006	0.003	0.007	-0.007	0.018	0.000	0.007	-0.004	0.008	0.005
	TDR	0.251	0.310	0.333	0.328	3.295	1.013	1.136	1.141	12.772	1.154	1.221	1.498	1.321	1.393	1.137
	NT	10	10	10	10	8	9	8	10	6	9	7	6	2	8	5
$\alpha=0.5$	IR	0.005	0.009	0.004	-0.028	0.078	-0.003	0.004	0.011	0.030	0.008	0.009	0.006	0.001	0.013	0.012
	TDR	0.244	0.235	0.370	0.178	3.123	1.035	1.047	1.007	12.613	1.087	1.267	1.410	1.419	1.277	1.443
	NT	7	9	9	5	9	5	8	6	8	8	7	6	6	6	7

**Table 8** Resolution methods comparison in the case of  $V(Wb, Wb, Wb, Wb)$

	$p_j=0.2$ (10 jobs)										$p_j=0.5$ (24 jobs)										$p_j=0.8$ (39 jobs)										$p_j=1$ (48 jobs)									
	H1*		H2*		H3*		H4*		B-MILP		H1*		H2*		H3*		H4*		B-MILP		H1*		H2*		H3*		H4*		B-MILP		H1*		H2*		H3*		H4*		B-MILP	
$\alpha=1$	$f_2$	<b>228.5</b>	<b>228.5</b>	228.6	<b>228.5</b>	228.5	<b>228.5</b>	882.7	<b>842.4</b>	857.55	<b>842.4</b>	878.5	878.5	<b>842.4</b>	878.5	2481	2397.2	2406	<b>2360</b>	-	4051.6	3900.7	3861.5	<b>3844</b>	-															
	MDI (s)	0.02	0.02	0.02	0.02	0.02	0.02	0.10	0.09	0.09	0.10	53.11	53.11	0.10	0.10	0.83	0.71	0.71	0.66	-	1.50	1.45	1.23	1.28	-															
	Time (s)	0.27	0.27	0.29	0.30	1.60	1.60	1.15	1.27	1.25	1.29	144.76	144.76	1.29	1.29	9.50	8.59	8.64	8.88	-	22.44	21.97	20.25	20.46	-															
$\alpha=0.75$	$f_2$	181.9	<b>181.8</b>	181.9	<b>181.8</b>	<b>181.8</b>	731.4	<b>704.9</b>	709.3	713.9	723.1	723.1	723.1	<b>2002.4</b>	2037.3	2164.7	<b>2002.4</b>	2037.3	2031.1	-	3532.3	<b>3217.4</b>	3303.8	3296.6	-															
	MDI (s)	0.02	0.02	0.02	0.02	0.27	0.27	0.09	0.09	0.09	0.10	497.57	497.57	0.10	0.68	0.64	0.64	0.66	0.68	-	1.38	1.41	1.44	1.52	-															
	Time (s)	0.25	0.27	0.27	0.27	1.64	1.64	1.22	1.24	1.24	1.23	787.78	787.78	1.23	8.35	8.14	8.14	8.46	9.12	-	21.00	20.69	22.01	19.93	-															
$\alpha=0.5$	$f_2$	134.1	<b>134.0</b>	134.1	<b>134.0</b>	<b>134.0</b>	539.6	<b>493.2</b>	513.8	509	497.0	497.0	497.0	1694	<b>1359.2</b>	1425	1430.2	1425	1430.2	-	2754.5	<b>2093.2</b>	2271.1	2254.7	-															
	MDI (s)	0.02	0.02	0.02	0.02	0.23	0.23	0.10	0.09	0.09	0.12	185.11	185.11	0.65	0.61	0.68	0.65	0.68	0.65	-	1.41	1.38	1.37	1.48	-															
	Time (s)	0.26	0.27	0.28	0.27	1.59	1.59	1.22	1.27	1.21	1.24	370.73	370.73	1.24	8.67	8.14	8.14	9.01	8.74	-	22.17	20.50	20.68	22.62	-															

The values highlighted in bold represent the optimal outcomes achieved in each iteration *B-MILP* the method based on the MILP, *MDI*(s) maximum duration of iteration in second, *Time*(s) the computing time of all iterations in second

**Table 9** Percentage error and standard deviation with the best solution in the case of  $V(Wb, Wb, Wb, Wb)$

	$p_\beta=0.2$ (10 jobs)					$p_\beta=0.5$ (24 jobs)					$p_\beta=0.8$ (39 jobs)					$p_\beta=1$ (48 jobs)						
	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP		
$\alpha=1$																						
Percentage error	<b>0.00</b>	<b>0.00</b>	0.01	<b>0.00</b>	<b>0.00</b>	5.28	<b>0.62</b>	2.34	<b>0.62</b>	5.01	5.44	1.89	2.27	<b>0.31</b>	–	5.82	1.91	0.86	<b>0.44</b>	–		
Standard deviation	0.00	0.00	0.07	0.00	0.00	4.20	1.70	2.88	1.70	3.27	2.39	1.26	4.10	0.74	–	2.00	1.73	0.90	1.01	–		
No. of best solution	50.00	50.00	47.00	50.00	50.00	5.00	40.00	10.00	40.00	3.00	5.00	4.00	20.00	35.00	–	0.00	4.00	15.00	35.00	–		
$\alpha=0.75$																						
Percentage error	0.02	<b>0.00</b>	0.03	<b>0.00</b>	<b>0.00</b>	5.93	<b>2.39</b>	2.89	3.47	4.91	8.79	<b>0.67</b>	2.38	2.08	–	9.92	<b>0.21</b>	2.83	2.63	–		
Standard deviation	0.10	0.00	0.11	0.00	0.00	4.22	4.65	3.40	3.90	4.11	2.38	1.31	3.93	4.03	–	3.06	0.60	2.19	2.46	–		
No. of best solution	41.00	50.00	45.00	50.00	50.00	10.00	20.00	10.00	20.00	2.00	0.00	35.00	5.00	10.00	–	0.00	43.00	2.00	5.00	–		
$\alpha=0.5$																						
Percentage error	0.11	<b>0.00</b>	0.02	<b>0.00</b>	<b>0.00</b>	11.39	<b>1.89</b>	6.12	5.14	2.70	24.86	<b>0.09</b>	5.07	5.35	–	31.61	<b>0.01</b>	8.50	7.71	–		
Standard deviation	0.33	0.00	0.15	0.00	0.00	7.68	2.11	5.28	4.23	2.59	3.51	0.28	3.69	5.97	–	4.15	0.07	3.36	3.79	–		
No. of best solution	41.00	50.00	48.00	50.00	50.00	3.00	20.00	10.00	15.00	10.00	0.00	45.00	0.00	5.00	–	0.00	48.00	2.00	0.00	–		

The values highlighted in bold represent the optimal outcomes

### 5.1.2 Resolution methods comparison

We compared for the 50 generated instances, four improved heuristics with the B-MILP in terms of solution quality and computing time. Averages are presented in Tables 8 and 9.

As can be seen, when  $p_\beta$  increases, the number of arrived jobs at each period increase too, the problem then will be difficult to solve in a reasonable time with the B-MILP. Only the heuristics can provide solutions when  $p_\beta$  exceeds 0.5. The Maximum Duration of Iteration (MDI) is the maximum computation duration that the method takes, at each iteration, to provide a solution. It estimates the period between the occurrence of a job and the establishment of the new schedule. As shown in Table 8, the MDI of the B-MILP is large when  $p_\beta=0.5$ .

When  $p_\beta=0.2$ , as there are few disruptions, the heuristics and B-MILP provide close solutions. In this case, the percentage errors and standard deviations presented in Table 9 becomes small. The percentage error is the difference between the best solution and provided solution, as a percentage of the best solution.

$$\text{percentage error} = \frac{\text{provided solution} - \text{best solution}}{\text{best solution}} \times 100$$

When  $p_\beta=0.5$ , as the disruptions are medium, the B-MILP takes more time to provide a solution compared to the heuristics. The method based on MILP (B-MILP) is also a heuristic consisting in generating at each disruption the MILP for solving the problem. The B-MILP method may sometimes produce inferior solutions compared to heuristics. This occurs because the B-MILP method finds a locally optimal solution, which may not be the optimal solution for the entire problem (when all the jobs have appeared). In the predictive–reactive strategy, which is an iterative process, the solution obtained with B-MILP is used as a basis for solving the next problem in the subsequent iteration. However, in this process, all the jobs that have already started their execution before time  $t$  are excluded from the set of jobs. So, B-MILP and heuristics schedule a different set of jobs. Thus, the obtained solutions are different, sometimes in favor of the heuristics.

In dynamic environments, decision-makers need to establish a new plan after each disruption. This planning process should be initiated as quickly as possible, preferably before the occurrence of another disruption. In our case, based on the discretization assumption, we have made, a new job can potentially arrive at every time interval of  $\Delta t$ . Therefore, if the Maximum Duration of Iteration (MDI) exceeds  $\Delta t$ , it is considered as an unacceptable time frame. In our experiments, we have assumed that  $\Delta t$  is equivalent to 10 min, which is equal to 600 s. Therefore, if the MDI exceeds 600 s, it is considered as an unreasonable duration. However, we only interrupt the simulations after 12 h, regardless of the MDI exceeding 600 s. With  $p_\beta=0.8$  and 1, the B-MILP fails to give a solution in a reasonable time. Heuristics quickly provide solutions, generally the four heuristics are close in terms of computing time.

H4\* considers a list of jobs sequenced by descending order of  $w_j$  and use NEH method to reschedule this list of jobs. It provides better solutions when  $\alpha=1$  since the schedule stability is not considered. The jobs weights are very influential in this case. H2\* considers a list of jobs sequenced by the order obtained in the previous schedule and uses NEH method to reschedule this list of jobs. It provides better solutions when  $\alpha=0.75$ . As the schedule stability is considered in this case, the previous sequence is often maintained, and the new arrived job is often placed in the last positions, depending on its weight. This is already the principle of H2\*, which explains its adaptation to this case. When  $\alpha=0.5$ , the schedule stability is more considered. According to Table 9, the number of times that H2\* provides the best solution becomes very high. This approves the superiority of H2\* when the schedule stability is more considered. H1\* only puts the new job in all possible positions without using NEH method. In most cases, it provides a bad solution than others since H2\*, H3\* and H4\* are improved versions of H1\*.

As a conclusion, one of the best choices that the decision-maker can do is to use, for each case, a heuristic among the proposed heuristics. Ideally, H4\* when  $\alpha=1$ , and H2\* when  $\alpha=0.5$  or  $\alpha=0.75$ .

## 5.2 The case of $V(Wb, RSb, RCb^*, RCb)$

A vector of  $V(Wb, RSb, RCb^*, RCb)$  is studied in this sub-section, and the average results of the 50 different instances are presented in Tables 10 and 11.

When the blocking constraints are simultaneously mixed in one production system, the space of feasible solutions reduces since there are many constraints to satisfy at the same time. As can be seen in Table 10, the computing time and the MDI becomes large compared to the case without blocking constraints. The B-MILP can hardly provide solutions when  $p_\beta=0.5$ .

The B-MILP and the heuristics still give very close results when  $p_\beta=0.2$ . The results differ when the system is subjected to more disruptions,  $p_\beta>0.2$ . Also, according to Table 10, we always observe a superiority of H4\* when  $\alpha=1$ , and H2\* when  $\alpha=0.75$  and  $\alpha=0.5$ . Therefore, the interpretation established in Sect. 5.1 about the impact of  $\alpha$  on the solutions performance remains still correct.

However, a diminution of the percentage errors and standard deviations is observed in Table 11, in comparison with Table 9. As the mixed blocking constraints are used in this example, the space of feasible solutions reduces. Thus, the heuristic solutions become close to each other. H2\*, H3\*, H4\* converge in most of the cases toward the same solution, compared with the case without blocking. H1\* which does not use NEH method, has a higher percentage error, compared to other heuristics.

The constraint  $RCb^*$  describes the case where a machine is blocked by a job, until this one will finish on the following machine. According to Sauvey et al (2020), this blocking constraint links together two machines around the same job since it considers at least the next operation to schedule an in-course operation. Therefore, in the next sub-section we consider  $RCb^*$  introduced between the first and the second



**Table 10** Resolution methods comparison in the case of  $V(Wb, RSb, RCb^*, RCb)$

	$p_p=0.2$ (10 jobs)										$p_p=0.5$ (24 jobs)										$p_p=0.8$ (39 jobs)										$p_p=1$ (48 jobs)																			
	H1*					H2*					H3*					H4*					B-MILP					H1*					H2*					H3*					H4*					B-MILP				
	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP															
$\alpha=1$	$f_2$	614.7	614.7	614.7	<b>612.0</b>	612.0	612.0	612.0	612.0	612.0	2341.2	<b>2311.4</b>	2337.4	<b>2311.4</b>	2325.3	2325.3	2325.3	2325.3	2325.3	2325.3	5628.4	5610.1	5610.1	5626.1	<b>5604.8</b>	5604.8	5604.8	5604.8	5604.8	8476.8	8353.6	8353.6	8355.9	<b>8338.3</b>	8338.3	8338.3	8338.3	8338.3	1.37	1.14	1.14	1.19	1.18	1.18	1.18	1.18	1.18			
	MDI	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.12	0.11	0.11	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.93	0.83	0.83	0.87	0.87	0.87	0.87	0.87	0.87	1.37	1.14	1.14	1.19	1.18	1.18	1.18	1.18	1.18	1.37	1.14	1.14	1.19	1.18	1.18	1.18	1.18	1.18			
	Time	0.24	0.24	0.24	0.23	1.68	1.48	1.38	1.43	1.45	4138.47	11.95	11.12	10.98	11.74	11.74	11.74	11.74	11.74	11.74	11.95	11.12	10.98	11.74	11.74	11.74	11.74	11.74	11.74	20.21	18.33	18.41	19.35	19.35	19.35	19.35	19.35	19.35	20.21	18.33	18.41	19.35	19.35	19.35	19.35	19.35	19.35			
$\alpha=0.75$	$f_2$	469.7	<b>467.1</b>	467.5	467.5	467.4	1842.7	<b>1836.9</b>	1839.5	1843.1	1843.8	4707.1	<b>4636.8</b>	4640.3	4642.8	4642.8	4642.8	4642.8	4642.8	4642.8	0.86	0.79	0.83	0.86	0.86	0.86	0.86	0.86	0.86	7036.8	<b>6877.3</b>	6877.4	6886.5	6886.5	6886.5	6886.5	6886.5	6886.5	1.22	1.16	1.21	1.20	1.20	1.20	1.20	1.20	1.20			
	MDI	0.02	0.02	0.02	0.02	0.23	0.12	0.11	0.12	0.14	593.00	0.86	0.79	0.83	0.86	0.86	0.86	0.86	0.86	0.86	0.86	0.79	0.83	0.86	0.86	0.86	0.86	0.86	0.86	1.22	1.16	1.21	1.20	1.20	1.20	1.20	1.20	1.20	1.22	1.16	1.21	1.20	1.20	1.20	1.20	1.20	1.20			
	Time	0.25	0.25	0.25	0.24	1.58	1.45	1.46	1.52	1.58	1697.34	11.01	10.64	11.06	11.54	11.54	11.54	11.54	11.54	11.54	11.01	10.64	11.06	11.54	11.54	11.54	11.54	11.54	11.54	18.93	18.24	18.70	18.83	18.83	18.83	18.83	18.83	18.83	18.93	18.24	18.70	18.83	18.83	18.83	18.83	18.83	18.83			
$\alpha=0.5$	$f_2$	320.6	<b>319.0</b>	319.3	319.8	319.8	1379.1	<b>1301.8</b>	1312.3	1310.3	1308.8	3553.7	<b>3246.1</b>	3283.9	3312.6	3312.6	3312.6	3312.6	3312.6	3312.6	0.85	0.89	0.87	0.87	0.87	0.87	0.87	0.87	0.87	5551.8	<b>4803.3</b>	4849.5	4824.3	4824.3	4824.3	4824.3	4824.3	4824.3	1.27	1.19	1.17	1.19	1.19	1.19	1.19	1.19	1.19			
	MDI	0.02	0.02	0.02	0.02	0.23	0.10	0.11	0.11	0.10	1071.68	0.85	0.89	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.85	0.89	0.87	0.87	0.87	0.87	0.87	0.87	0.87	1.27	1.19	1.17	1.19	1.19	1.19	1.19	1.19	1.19	1.27	1.19	1.17	1.19	1.19	1.19	1.19	1.19	1.19			
	Time	0.26	0.24	0.24	0.24	32.78	1.35	1.35	1.37	1.44	1764.43	11.67	11.26	11.31	11.23	11.23	11.23	11.23	11.23	11.23	11.67	11.26	11.31	11.23	11.23	11.23	11.23	11.23	11.23	20.47	18.71	19.12	18.55	18.55	18.55	18.55	18.55	18.55	20.47	18.71	19.12	18.55	18.55	18.55	18.55	18.55	18.55			

The values highlighted in bold represent the optimal outcomes

**Table 11** Percentage error and standard deviation in the case of  $V(Wb, RSb, RCb^*, RCb)$

	$p_\beta=0.2$ (10 jobs)					$p_\beta=0.5$ (24 jobs)					$p_\beta=0.8$ (39 jobs)					$p_\beta=1$ (48 jobs)							
	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP			
$\alpha=1$																							
Percentage error	0.47	0.47	0.47	<b>0.00</b>	<b>0.00</b>	1.36	<b>0.03</b>	1.18	<b>0.03</b>	0.65	0.53	0.23	0.49	<b>0.14</b>	-	1.61	0.19	0.23	<b>0.01</b>	-			
Standard deviation	1.48	1.48	1.48	0.00	0.00	1.12	0.05	2.19	0.05	0.86	1.19	0.33	0.63	0.23	-	1.89	0.23	0.49	0.03	-			
No. of best solution	45	45	45	50	50	10	35	20	35	20	15	25	25	30	-	0	25	35	40	-			
$\alpha=0.75$																							
Percentage error	0.57	<b>0.00</b>	0.09	0.09	0.08	0.58	<b>0.22</b>	0.38	0.59	0.61	1.71	<b>0.23</b>	0.30	0.35	-	2.37	<b>0.07</b>	0.07	0.20	-			
Standard deviation	0.84	0.00	0.30	0.30	0.24	1.14	0.53	0.63	0.81	0.89	1.39	0.34	0.40	0.37	-	1.04	0.15	0.08	0.21	-			
No. of best solution	30	50	45	45	45	20	30	25	20	25	15	25	5	5	-	0	40	20	15	-			
$\alpha=0.5$																							
Percentage error	0.59	<b>0.07</b>	0.19	0.19	0.34	6.84	<b>0.80</b>	1.61	1.44	1.29	11.08	<b>1.43</b>	2.62	3.48	-	16.14	<b>0.28</b>	1.29	0.73	-			
Standard deviation	0.87	0.21	0.42	0.42	0.60	3.73	1.78	1.34	1.17	0.91	5.53	3.74	3.52	4.52	-	5.10	0.40	0.81	0.91	-			
No. of best solution	20	45	40	40	35	0	40	5	5	0	5	30	0	15	-	0	25	5	20	-			

**Table 12** Resolution methods comparison in the case of  $V(RCb^*, Wb, Wb, R Sb)$

	$p_p=0.2$ (10 jobs)										$p_p=0.5$ (24 jobs)										$p_p=0.8$ (39 jobs)										$p_p=1$ (48 jobs)																			
	$H1^*$					$H2^*$					$H3^*$					$H4^*$					$B-MILP$					$H1^*$					$H2^*$					$H3^*$					$H4^*$					$B-MILP$				
	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$	$H1^*$	$H2^*$	$H3^*$	$H4^*$	$B-MILP$																				
$\alpha=1$	<b>443.6</b>	<b>443.6</b>	<b>443.6</b>	<b>443.6</b>	—	1876.4	1847.0	1870.0	<b>1844.6</b>	—	5183.8	5166.0	5178.7	<b>5165.1</b>	—	8218.3	8198.6	8198.6	8198.1	<b>8064.0</b>	—	2.15	2.02	1.92	2.12	—	2.15	2.02	1.92	2.12	—																			
MDI (s)	0.04	0.05	0.04	0.04	3.04	0.28	0.26	0.32	0.29	—	1.32	1.10	1.13	1.13	—	1.32	1.10	1.13	1.13	—	1.32	1.10	1.13	1.13	—	1.32	1.10	1.13	1.13	—																				
Time (s)	0.46	0.42	0.43	0.41	11.92	2.73	2.90	3.05	2.93	—	15.75	14.64	14.37	14.49	—	32.61	31.35	29.98	30.52	—	32.61	31.35	29.98	30.52	—	32.61	31.35	29.98	30.52	—																				
$\alpha=0.75$	<b>385.5</b>	<b>383.5</b>	383.6	383.6	<b>383.5</b>	1671.6	<b>1627.3</b>	1631.6	1632.1	—	4574.7	<b>4398.8</b>	4444.4	4421.8	—	7130.1	<b>6841.5</b>	6930.4	6898.4	—	7130.1	<b>6841.5</b>	6930.4	6898.4	—	7130.1	<b>6841.5</b>	6930.4	6898.4	—																				
MDI (s)	0.04	0.04	0.04	0.04	3.73	0.25	0.30	0.29	0.28	—	1.14	1.18	1.12	1.17	—	2.32	1.84	2.03	2.16	—	2.32	1.84	2.03	2.16	—	2.32	1.84	2.03	2.16	—																				
Time (s)	0.40	0.42	0.43	0.44	12.92	2.85	2.98	3.03	2.95	—	14.82	14.98	14.85	14.85	—	33.48	29.83	30.68	31.13	—	33.48	29.83	30.68	31.13	—	33.48	29.83	30.68	31.13	—																				
$\alpha=0.5$	<b>310.1</b>	<b>286.5</b>	287.7	286.6	286.9	1393.1	<b>1166.7</b>	1188.2	1186.1	—	3689.0	<b>2948.1</b>	3052.4	2997.8	—	5646.8	<b>4462.3</b>	4558.9	4533.4	—	5646.8	<b>4462.3</b>	4558.9	4533.4	—	5646.8	<b>4462.3</b>	4558.9	4533.4	—																				
MDI (s)	0.04	0.04	0.04	0.04	3.82	0.33	0.27	0.30	0.30	—	1.16	1.12	1.15	1.04	—	2.48	2.11	1.89	2.14	—	2.48	2.11	1.89	2.14	—	2.48	2.11	1.89	2.14	—																				
Time (s)	0.43	0.41	0.43	0.43	13.64	3.32	3.03	3.23	3.08	—	15.41	14.54	15.66	14.46	—	35.74	31.71	29.78	31.67	—	35.74	31.71	29.78	31.67	—	35.74	31.71	29.78	31.67	—																				

The values highlighted in bold represent the optimal outcomes

**Table 13** Percentage error and standard deviation in the case of  $V(RCb^*, Wb, Wb, RSb)$

	$p_\beta = 0.2$ (10 jobs)					$p_\beta = 0.5$ (24 jobs)					$p_\beta = 0.8$ (39 jobs)					$p_\beta = 1$ (48 jobs)						
	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP	H1*	H2*	H3*	H4*	B-MILP		
$\alpha = 1$																						
Percentage error	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	2.13	0.26	1.78	<b>0.12</b>	–	0.40	0.06	0.30	<b>0.04</b>	–	2.18	1.95	1.94	<b>0.02</b>	–		
Standard deviation	0.00	0.00	0.00	0.00	0.00	4.95	0.26	4.78	0.21	–	0.20	0.10	0.24	0.10	–	5.32	5.38	5.24	0.05	–		
No. of best solution	50	50	50	50	50	5	15	25	30	–	0	30	5	40	–	5	10	10	25	–		
$\alpha = 0.75$																						
Percentage error	0.56	<b>0.04</b>	0.05	0.05	<b>0.04</b>	2.82	<b>0.09</b>	0.34	0.38	–	4.04	<b>0.00</b>	1.03	0.52	–	4.24	<b>0.00</b>	1.28	0.80	–		
Standard deviation	0.71	0.12	0.12	0.12	0.12	0.45	0.14	0.55	0.53	–	0.90	0.00	0.74	0.59	–	0.59	0.00	0.81	0.70	–		
No. of best solution	30	40	40	40	40	0	35	30	15	–	0	50	5	10	–	0	45	5	5	–		
$\alpha = 0.5$																						
Percentage error	8.48	<b>0.09</b>	0.59	0.10	0.24	20.14	<b>0.02</b>	1.92	1.74	–	25.72	<b>0.01</b>	3.48	1.77	–	27.07	<b>0.10</b>	2.32	1.77	–		
Standard deviation	5.18	0.28	0.91	0.30	0.49	6.76	0.07	1.05	1.22	–	6.44	0.04	5.14	1.31	–	5.40	0.21	1.57	1.78	–		
No. of best solution	7	45	35	45	40	0	45	5	5	–	0	45	5	0	–	0	30	5	15	–		

The values highlighted in bold represent the optimal outcomes

machine, followed by two successive  $Wb$ . This, for evaluating the impact of this particular situation on the computing time and the percentage errors of the proposed methods.

### 5.3 The case of $V(RCb^*, Wb, Wb, RSb)$

The vector  $V(RCb^*, Wb, Wb, RSb)$  is studied in this sub-section, the average results of the 50 instances are presented in Tables 12 and 13.

Having  $RCb^*$  constraint introduced between the first and the second machine, makes the problem difficult to solve. Since this blocking considers the next operation to schedule an in-course one, the B-MILP needs more time to find the best solution. Therefore, the B-MILP can only provide solutions when the system is subjected to little disruptions. When  $p_\beta$  exceeds 0.2, it fails to give solutions in a reasonable time. The simulation has been interrupted after 12 h. Heuristics still provide solutions in a short time. However, the computing time and the MDI are long compared to the blocking case previously studied.

On the other hand, a diminution of the percentage errors and standard deviations is observed in Table 13, in comparison with Table 11. As the  $RCb^*$  is introduced between the first and the second machine, the space of feasible solutions becomes even smaller.  $H2^*$ ,  $H3^*$ , and  $H4^*$  are based on NEH method for rescheduling jobs, converge even more toward the same solution.  $H1^*$  has a large percentage error compared to other heuristics. Since it only puts the new job in all possible positions without using NEH method, it fails in most of cases to provide the best solution.

In this particular blocking case,  $H4^*$  is still efficient when the stability is not considered ( $\alpha = 1$ ).  $H2^*$  is also still efficient when the schedule stability is more considered.  $H3^*$  provides better results compared to the blocking cases previously studied.

## 6 Conclusion and perspectives

This paper investigates a flow shop rescheduling problem when different blocking constraints are mixed in one production system. Two aspects are simultaneously investigated to measure the schedule performance. First, regarding the efficiency, the total weighted waiting time is considered as a criterion. Second, in terms of stability, the total weighted completion time deviation is used as a criterion to limit the difference from the initial schedule. At each period, the established schedule can be disrupted by the arrival of a new job. Using the predictive–reactive strategy, the schedule is updated in response to this disruption. The problem has first been solved using a MILP model. Experimental results show that the MILP resolution is only possible for little size instances. Hence, inspired by NEH algorithm, we proposed four heuristics for solving large size instances of this problem. The comparison of these methods has been discussed in the experimental results section, where three

different blocking constraints vectors have been evaluated. The main conclusions of this work are:

- The appearance frequency of the jobs  $p_\beta$  has a major impact on the computing time. The more arriving jobs increase, the more the resulting computing time increases. The increase in computing time depends also on the type of successive blocking constraints between the machines.
- When the schedule stability is not considered, improved Heuristic 4 (H4\*) which sequences the set of jobs by descending order of  $w_j$  and uses NEH method for rescheduling, provides better solutions since the jobs weights have a major impact on the results. However, when the schedule stability is more considered, improved Heuristic 2 (H2\*) which consists in maintaining the previous order and in using NEH method for rescheduling, gives better solutions since the deviation from the previous schedule is limited by the stability criterion. One of the best choices that the decision-maker can do, is to use, for each case, a heuristic among the proposed heuristics. Ideally, H4\* when  $\alpha = 1$ , and H2\* when  $\alpha = 0.5$  or  $\alpha = 0.75$ .
- Considering blocking constraints mixed in one flow shop system, reduces the space of feasible solutions since there are many constraints to satisfy. This increases the resolution time, but it reduces the percentage errors of the resolution methods. The B-MILP and the proposed heuristics converge in most of the cases toward the same solution. This matter has been clearly shown when  $RCb^*$  constraint has been considered as a first constraint in the flow shop system.

This work can be of great interest, not only for the researchers facing with flow shop rescheduling problems, but also for a broader audience, such as industrial decision-makers. As perspectives, we intend to improve our methods by adapting it to consider, at each period, more than one job. Indeed, the decision-makers must quickly react for providing a new schedule in response to disruptions, even if several ones arrive at the same time. On the other hand, the proposed heuristics accelerate the getting of solutions. However, the decision-makers must choose among the proposed heuristics, the best adapted to their case, depending on the efficiency-stability coefficient value. In future work, it will be interesting to design a metaheuristic that can be smartly adapted to all cases. Finally, we will study the behavior of the *TWWT* combined with the *TWCTD* in the case of job shop or open shop environments, considering the blocking constraints.

**Acknowledgements** This work is supported by the Urban Community of Sarreguemines-France and the Grand Est Region-France.

**Data availability** The data that support the findings of this study are available from the corresponding author, AT, upon reasonable request.

## Declarations

**Conflict of interest** The authors declare no conflict of interest.

## References

- Akkan C (2015) Improving schedule stability in single-machine rescheduling for new operation insertion. *Comput Oper Res* 64:198–209
- Akyol Ozer E, Sarac T (2019) MIP models and a metaheuristic algorithm for an identical parallel machine scheduling problem under multiple copies of shared resources constraints. *TOP* 27(1):94–124
- Auer P, Dósa G, Dulai T, Fügenschuh A, Näser P, Ortner R, Werner-Stark Á (2021) A new heuristic and an exact approach for a production planning problem. *CEJOR* 29(3):1079–1113
- Bautista-Valhondo J, Alfaro-Pozo R (2020) Mixed integer linear programming models for Flow Shop Scheduling with a demand plan of job types. *CEJOR* 28(1):5–23
- Braune R, Gutjahr WJ, Vogl P (2022) Stochastic radiotherapy appointment scheduling. *CEJOR* 30(4):1239–1277
- De La Vega J, Moreno A, Morabito R, Munari P (2023) A robust optimization approach for the unrelated parallel machine scheduling problem. *TOP* 31(1):31–66
- Druetto A, Pastore E, Renier E (2022) Parallel batching with multi-size jobs and incompatible job families. *Top* 31(2):440–458
- Guo Y, Xie X (2017) Two mixed integer programming formulations on single machine to reschedule repaired jobs for minimizing the total waiting-time. Chinese Automation Congress (CAC), IEEE, pp 2129–2133
- Gürel S, Körpeoğlu E, Aktürk MS (2010) An anticipative scheduling approach with controllable processing times. *Comput Oper Res* 37(6):1002–1013
- Hall NG, Potts CN (2004) Rescheduling for new orders. *Oper Res* 52(3):440–453
- Haroune M, Dhib C, Neron E, Soukhal A, Mohamed Babou H, Nanne MF (2022) Multi-project scheduling problem under shared multi-skill resource constraints. *Top* 31(1):194–235
- He X, Dong S, Zhao N (2020) Research on rush order insertion rescheduling problem under hybrid flow shop based on NSGA-III. *International journal of production research* 58(4):1161–1177
- Herrmann JW (2006) Rescheduling strategies, policies, and methods. *Handbook of production scheduling*. Springer, Boston, pp 135–148
- Kacem A, Dammak A (2021) Multi-objective scheduling on two dedicated processors. *TOP* 29(3):694–721
- Kan AR (1976) Problem formulation. *Machine scheduling problems*. Springer, Boston, pp 5–29
- Katragjini K, Vallada E, Ruiz R (2013) Flow shop rescheduling under different types of disruption. *Int J Prod Res* 51(3):780–797
- Kecman P, Corman F, D'Ariano A, Goverde RM (2013) Rescheduling models for railway traffic management in large-scale networks. *Public Transp* 5(1–2):95–123
- Kovalyov MY, Kress D, Meiswinkel S, Pesch E (2019) A parallel machine schedule updating game with compensations and clients averse to uncertain loss. *Comput Oper Res* 103:148–157
- Li Y, Carabelli S, Fadda E, Manerba D, Tadei R, Terzo O (2020) Machine learning and optimization for production rescheduling in Industry 4.0. *Int J Adv Manuf Technol* 110(9):2445–2463
- Liu Z, Ro YK (2014) Rescheduling for machine disruption to minimize makespan and maximum lateness. *J Sched* 17(4):339–352
- Liu L, Zhou H (2013) Open shop rescheduling under singular machine disruption. *Comput Integr Manuf Syst* 10:12
- Lodree E Jr, Jang W, Klein CM (2004) A new rule for minimizing the number of tardy jobs in dynamic flow shops. *Eur J Oper Res* 159(1):258–263
- Machado-Dominguez LF, Paternina-Arboleda CD, Vélez JI, Barrios-Sarmiento A (2022) An adaptive bacterial foraging optimization algorithm for solving the MRCPSP with discounted cash flows. *TOP* 30(2):221–248
- Manzini M, Demeulemeester E, Urgo M (2022) A predictive–reactive approach for the sequencing of assembly operations in an automated assembly line. *Robot Comput-Integr Manuf* 73:102201
- Martinez S, Dauzère-Pérès S, Gueret C, Mati Y, Sauer N (2006) Complexity of flowshop scheduling problems with a new blocking constraint. *Eur J Oper Res* 169(3):855–864
- Mohan J, Lanka K, Rao NA, Manupati VK (2022) Sustainable flexible job shop scheduling: a systematic literature review. In: *Global congress on manufacturing and management*. Springer, Cham, pp 227–246
- Mula J, Bogataj M (2021) OR in the industrial engineering of Industry 4.0: experiences from the Iberian Peninsula mirrored in CJOR. *Central Eur J Oper Res* 29(4):1163–1184

- Nawaz M, Ensfore EE Jr, Ham I (1983) A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega* 11(1):91–95
- Ozolins A (2021) Dynamic programming approach for solving the open shop problem. *CEJOR* 29(1):291–306
- Pitombeira-Neto AR, Prata BDA (2020) A matheuristic algorithm for the one-dimensional cutting stock and scheduling problem with heterogeneous orders. *TOP* 28(1):178–192
- Prata BDA, de Abreu LR, Lima JYF (2021) Heuristic methods for the single-machine scheduling problem with periodical resource constraints. *TOP* 29(2):524–546
- Rahmani D, Ramezani R (2016) A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: a case study. *Comput Ind Eng* 98:360–372
- Sauvey C, Sauer N (2020) Two NEH heuristic improvements for flowshop scheduling problem with makespan criterion. *Algorithms* 13(5):112
- Sauvey C, Trabelsi W, Sauer N (2020) Mathematical model and evaluation function for conflict-free warranted makespan minimization of mixed blocking constraint job-shop problems. *Mathematics* 8(1):121
- Sayed SI, Contreras I, Diaz JA, Luna DE (2020) Integrated cross-dock door assignment and truck scheduling with handling times. *TOP* 28(3):705–727
- Şenyiğit E, Atici U, Şenol MB (2022) Effects of OCRA parameters and learning rate on machine scheduling. *Central Eur J Oper Res* 30:941–959
- Serrano-Ruiz JC, Mula J, Poler R (2021) Smart manufacturing scheduling: a literature review. *J Manuf Syst* 61:265–287
- Serrano-Ruiz JC, Mula J, Poler R (2022) Development of a multidimensional conceptual model for job shop smart manufacturing scheduling from the Industry 4.0 perspective. *J Manuf Syst* 63:185–202
- Tao Z, Liu X (2019) Dynamic scheduling of dual-resource constrained blocking job shop. In: *International conference on intelligent robotics and applications*. Springer, Cham, pp 447–456
- Tighazoui A, Sauvey C, Sauer N (2020) New efficiency-stability criterion in a rescheduling problem with dynamic jobs weights. In: *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, vol 1, IEEE, pp 475–480
- Tighazoui A, Sauvey C, Sauer N (2021a) Predictive-reactive strategy for flowshop rescheduling problem: minimizing the total weighted waiting times and instability. *J Syst Sci Syst Eng* 30:253–275
- Tighazoui A, Sauvey C, Sauer N (2021b) Predictive-reactive strategy for identical parallel machine rescheduling. *Comput Oper Res* 134:105372
- Trabelsi W, Sauvey C, Sauer N (2011) Complexity and mathematical model for flowshop problem subject to different types of blocking constraint. *IFAC Proc Vol* 44(1):8183–8188
- Trabelsi W, Sauvey C, Sauer N (2012) Heuristics and metaheuristics for mixed blocking constraints flowshop scheduling problems. *Comput Oper Res* 39(11):2520–2527
- Uhlmann IR, Zanella RM, Frazzon EM (2022) Hybrid flow shop rescheduling for contract manufacturing services. *Int J Prod Res* 60(3):1069–1085
- Valledor P, Gomez A, Puente J, Fernandez I (2022) Solving rescheduling problems in dynamic permutation flow shop environments with multiple objectives using the hybrid dynamic non-dominated sorting genetic II algorithm. *Mathematics* 10(14):2395
- Vieira GE, Herrmann JW, Lin E (2003) Rescheduling manufacturing systems: a framework of strategies, policies, and methods. *J Sched* 6(1):39–62
- Wang L, Zhang L, Zheng DZ (2006) An effective hybrid genetic algorithm for flow shop scheduling with limited buffers. *Comput Oper Res* 33(10):2960–2971
- Wu Q, Xie N, Zheng S, Bernard A (2022) Online order scheduling of multi 3D printing tasks based on the additive manufacturing cloud platform. *J Manuf Syst* 63:23–34
- Xiao J, Osterweil LJ, Wang Q, Li M (2010) *Dynamic resource scheduling in disruption-prone software development environments*. International conference on fundamental approaches to software engineering. Springer, Berlin, pp 107–122
- Yan P, Liu SQ, Sun T, Ma K (2018) A dynamic scheduling approach for optimizing the material handling operations in a robotic cell. *Comput Oper Res* 99:166–177
- Yuan K, Sauer N, Sauvey C (2009) Application of EM algorithm to hybrid flow shop scheduling problems with a special blocking. In: *2009 IEEE conference on emerging technologies & factory automation*, IEEE, pp 1–7. <https://doi.org/10.1109/ETFA.2009.5347066>
- Zhang L, Gao L, Li X (2013) A hybrid intelligent algorithm and rescheduling technique for job shop scheduling problems with disruptions. *Int J Adv Manuf Technol* 65(5):1141–1156



- Zhang L, Hu Y, Wang C, Tang Q, Li X (2022a) Effective dispatching rules mining based on near-optimal schedules in intelligent job shop environment. *J Manuf Syst* 63:424–438
- Zhang X, Lin WC, Wu CC (2022b) Rescheduling problems with allowing for the unexpected new jobs arrival. *J Comb Optim* 43(3):630–645

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.